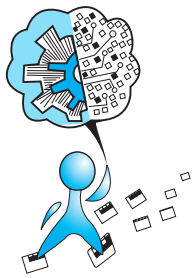

Algorithm: The essence of Computing Science

Tirimula Rao Benala

Faculty in the Dept. of Information Technology at JNTUGV, College of Engineering, Vizianagaram.
Email: btirimula.it@jntukucev.ac.in

Algorithms are step-by-step instructions for solving a particular problem. If your algorithm involves significant amounts of input data, complex operations, or both, you need to build clever algorithms that computers can process quickly. The myth is that we do not need algorithms for aspirants to become software developers and full-stack engineers. We can use existing face recognition algorithms, Dijkstra's shortest path algorithm, and external libraries to solve real-time problems. However, MAANG companies like Google use tailor-made shortest path algorithms to find the shortest path between source and destination for Google maps. Similarly, Meta uses sophisticated algorithms for friend recommendations on Facebook. Hence, we must adjust the existing library or build code from scratch. It is not correct to judge a candidate by technology or language ability. Therefore, big companies prefer asking algorithms questions rather than technology or language. Algorithms help test the quality of candidates. Adapt specific solutions to specific situations through new approaches to newly seen problems and confidently present solutions to teams with different perspectives.

In computer science, data (treated as singular, plural, or mass nouns) is any sequence of one or more symbols. Digital data is data represented by a binary number system of 1s and 0s rather than an analog representation. In modern computer systems (since 1960), all data is digital. Data representing quantities, characters, or symbols on which a computer performs operations are stored and recorded in magnetic, optical, electronic, or mechanical recording media and transmitted in the form of digital electrical or optical signals. Digital data is often stored in relational databases, such as spreadsheets and SQL databases, and can generally be represented as abstract key-value pairs. Data structures can store different data types, such as numbers, strings, and other data structures. In computer science, a data structure is a data organization, management, and storage format typically chosen for efficient access to data. More specifically, a data structure is a collection of data values, relationships between them, and functions or operations that can be applied to data. This is the algebraic structure of data (Wikipedia). Algorithm designers must choose appropriate data structures to design efficient algorithms.

Algorithm design is a method or mathematical process for problem-solving and algorithm development. Algorithm design is part of many solution theories in operations research, Dynamic programming, and the divide-and-conquer principle. Techniques for designing and implementing algorithmic designs are also known as algorithmic design patterns, and examples include the template-method pattern and the decorator pattern.

Resource efficiency is one of the most important considerations while designing algorithms (runtime, memory usage). Using the Big O notation, one may show how an algorithm's run-time grows as the size of its input does.

Typical procedures for creating algorithms:

1. issue description
2. creation of a model
3. Information about the algorithm
4. constructing an algorithm
5. examining the algorithm's accuracy
6. a review of the algorithm
7. application of an algorithm
8. program evaluation
9. preparing documentation

Algorithmic effectiveness [4]:

When developing a resolution to an issue, we frequently want to gauge how "quick" the solution is. Unfortunately, many factors influence how quickly your software runs. The hardware of your computer, the dataset you're using, other applications running on it, and even the temperature of your room can all impact how quickly your program runs. Thus, while using timers to assess program speed is helpful, it is not necessarily a valid technique to contrast different implementations.

Big-O notation: Context

Big-O notation is a more theoretical method of comparing different algorithms. Big-O notation is a mathematical notion that provides a rough upper bound on how long an algorithm will take to execute based on the size of the

dataset it uses. Read this article if you're curious about the notation's precise definition. For the time being, the emphasis is on making the notation simple to comprehend and efficient to apply.

Big-O Notation

It is helpful to know what is vital and what is not essential before using Big O notation. There are a few guidelines to follow while using the notation:

1. Constants are not considered
2. the term with the most rapid growth is considered

Considering those guidelines, the following statements are equivalent:

O(90 * n) → O(n)

O (0.231) → O(1)

$O(4*n^2 + 3n + 2) \rightarrow O(n^2)$

O(n*log(n) + 5000 * n) → O(n * log(n))

In each example, constant values are changed to 1, and other smaller components are not considered.

The mathematical definition of Big O is as follows:

Let f(n) and g(n) be positive functions, and let the set of functions be denoted by O(g(n)). f(n) is a member of the set O(g(n)) for which the ratio f(n)/g(n) remains below a constant c when n becomes very large.

Example #1: f(n) = 50n is O(n), as the ratio of f(n) to g(n) = n is always c=50.

Example #2: 4n-3 is also in O(n) because the ratio between f(n) = 4n - 3 f(n) = 3n2 and g(n) = n remains at or below c = 4.Z

Complexity Analysis [6]:

What is the Time Complexity of the summation of n numbers?

Programmer perspectives:

Input: **n**

Output: **1+2+3+...+n**

```
Algorithm #1: function summation(n) {
var sum = 0; ——→ 1 fundamental operation
for (var i = 1; i < n; i++) {——→ n iteration
  sum = sum+i; ——→ 2 fundamental operation * n
 }
 return sum; ——→ 1 fundamental operation
          }
```

Time Complexity: number of fundamental operations

Fundamental operations:

a+b, a-b, a×b, a÷b, a=b, return, bitwise operations, Comparision operations

Algorithm #1 has 2*n*+2 fundamental operations

Time Complexity: O(2n+2) implies O(n)

Algorithm thinking perspective:

Input: **n**

Output: **1+2+3+...+n**

```
Algorithm #2: function summation(n) {
                return n * (n+1)/2;  ←— 4 fundamental op.
              }
                                      doesn't depends on n
```

Four fundamental operations, independent of input size n, are required.

Consequently, the time complexity is: O (1)

Algorithm #1 has a time complexity of O(n), whereas Algorithm #2 has a time complexity of O (1).

Conclusion:

CRUD applications require no algorithmic knowledge. However, in order to implement an innovative concept, we must design complex algorithms by modifying existing algorithms or from scratch. Thus, algorithms play a crucial role in developing innovative, cutting-edge products such as google maps, Amazon's product recommendation system, and Netflix's movie recommendation system.

References

[1] Algorithm - Wikipedia. (2022, July 1). Retrieved September 10, 2022, from https://en.wikipedia.org/wiki/Algorithm

[2] Data (computer science) - Wikipedia. (2022, January 14). Retrieved September 10, 2022, from https://en.wikipedia.org/wiki/Data_(computer_science)

[3] Data structure - Wikipedia. (2018, November 6). Retrieved September 10, 2022, from https://en.wikipedia.org/wiki/Data_structure

[4] School, R. (n.d.). Free JavaScript Computer Science Course | Rithm School. Retrieved September 10, 2022, from https://www.rithmschool.com/courses/javascript-computer-science-fundamentals/introduction-to-big-o-notation.

[5] Learn Algorithm Fundamentals on Brilliant. (n.d.). Retrieved September 11, 2022, from https://brilliant.org/courses/computer-science-algorithms/the-speed-of-algorithms-2/big-o-formally/3/

[6] Big O Introduction. (n.d.). Retrieved September 11, 2022, from https://rithmschool.github.io/function-timer-demo/

About the Author

Tirimula Rao Benala obtained Ph.D. from JNTU Kakinada. He has been working as a faculty in the Department of Information Technology at JNTUGV, College of Engineering, Vizianagaram, for the last ten years. He has published about 35 refereed journal and conference papers. Currently, he is heading the department. He has total teaching experience of 20 years.